

(19)日本国特許庁 (J P)

(12) 公 開 特 許 公 報 (A)

(11)特許出願公開番号

特開平10-308797

(43)公開日 平成10年(1998)11月17日

| (51)Int.Cl. <sup>6</sup> | 識別記号  | F I                   |
|--------------------------|-------|-----------------------|
| H 0 4 L 29/14            |       | H 0 4 L 13/00 3 1 5 Z |
| G 0 6 F 13/00            | 3 5 3 | G 0 6 F 13/00 3 5 3 U |
| H 0 4 L 12/40            |       | H 0 4 L 11/00 3 2 0   |
| 29/06                    |       | 13/00 3 0 5 Z         |

審査請求 未請求 請求項の数7 O L (全 8 頁)

(21)出願番号 特願平9-119036

(22)出願日 平成9年(1997)5月9日

(71)出願人 000002185

ソニー株式会社

東京都品川区北品川6丁目7番35号

(72)発明者 飯島 祐子

東京都品川区北品川6丁目7番35号 ソニー株式会社内

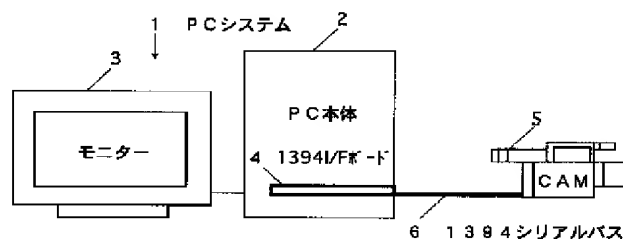
(74)代理人 弁理士 杉山 猛

(54)【発明の名称】 IEEE1394プロトコル検査方法及び装置

(57)【要約】

【課題】 IEEE1394高速シリアルバス対応の機器のプロトコルを自動的に検査する。

【解決手段】 PC本体2内のファイルメモリの内部には、検査対象であるCAM5が1394プロトコルを満たしているかどうかを検査するために必要な一連の送信パケットとそれに対応する受信アシンクロナスパケット、ACKコード、所望のメッセージを促すメッセージコード、所望の場所で待時間を入れるウェイトコード、所望の場所でルート／非ルートになるようにバスリセットを起こすことができるバスリセットコード等がデータファイルとして格納してある。そして、検査用アプリケーションを立ち上げ、処理を開始すると、このデータファイルが開かれ、以後各種パケットの送信、ACKの受信、エラーの検出等が自動的に実行される。



**【特許請求の範囲】**

**【請求項1】** IEEE1394高速シリアルバス対応の機器のプロトコルを検査する方法であって、少なくとも前記プロトコルの検査に必要な送信アシンクロナスパケットとそれに対応する受信アシンクロナスパケットをデータファイルとして保有する検査装置を用意し、この検査装置と前記機器を前記バスで接続し、前記検査装置の検査アプリケーション上で前記データファイルを開き、前記データファイルに書かれている送信アシンクロナスパケットを前記検査装置から前記機器へ前記バスを介して順次送信し、前記バスを介して前記機器から返ってきた受信アシンクロナスパケットと前記データファイルに書かれている前記受信アシンクロナスパケットを比較することを特徴とするIEEE1394プロトコル検査方法。

**【請求項2】** 前記機器から返ってきた受信アシンクロナスパケットが前記データファイルに書かれている前記受信アシンクロナスパケットと異なる場合には、前記機器から返ってきた受信アシンクロナスパケットのデータ及びエラー数のデータを保存する請求項1に記載のIEEE1394プロトコル検査方法。

**【請求項3】** 前記機器から返ってきた受信アシンクロナスパケットのデータ及びエラー数のデータをテキストファイルとして保存する請求項2に記載のIEEE1394プロトコル検査方法。

**【請求項4】** データファイルに誤りがあった場合には、データファイルにエラーがあることをメッセージ等でユーザーに知らせ、検査には入らない請求項3に記載のIEEE1394プロトコル検査方法。

**【請求項5】** IEEE1394高速シリアルバス対応の機器のプロトコルを検査する装置であって、少なくとも前記プロトコルの検査に必要な送信アシンクロナスパケットとそれに対応する受信アシンクロナスパケットを有するデータファイルを備え、検査アプリケーション上で前記データファイルを開き、前記データファイルに書かれている送信アシンクロナスパケットを前記機器へ前記バスを介して順次送信し、前記バスを介して前記機器から返ってきた受信アシンクロナスパケットと前記データファイルに書かれている前記受信アシンクロナスパケットを比較することを特徴とするIEEE1394プロトコル検査装置。

**【請求項6】** 前記データファイルには、さらにアックコード、所望のメッセージを促すメッセージコード、所望の場所で待時間を入れるウェイトコード、所望の場所でルート／非ルートになるようにバスリセットを起こすことができるバスリセットコードが書かれている請求項5に記載のIEEE1394プロトコル検査装置。

**【請求項7】** 前記データファイルを書き換えるだけで、実行ファイルは書き換えなくても検査を行うことができる請求項5に記載のIEEE1394プロトコル検査装置。

査装置。

**【発明の詳細な説明】****【0001】**

**【発明の属する技術分野】** 本発明は、IEEE1394高速シリアルバス対応の機器を開発する際に、IEEE1394プロトコルを満たしているかどうかを検査するための方法及び装置に関する。

**【0002】**

**【従来の技術】** IEEE1394高速シリアルバス（以下1394シリアルバスという）を用いてデジタルビデオ信号及びデジタルオーディオ信号の送受信を行う機能を備えたデジタルビデオカメラが既に商品化されている。また、パーソナルコンピュータ（以下PCという）に周辺装置を接続するインターフェースとして1394シリアルバスが注目されている。

**【0003】** 1394シリアルバスで接続した機器の間では、デジタルビデオ信号及びデジタルオーディオ信号等のリアルタイムデータのアイソクロナス伝送と、各種制御コマンド等のアシンクロナス伝送を行うことができる。

**【0004】** 従来、1394シリアルバス対応の機器がIEEE1394プロトコル（以下1394プロトコルという）を満たしているかどうかを検査する際には、PC等で構成した検査装置と、検査対象機器との間を1394シリアルバスで接続する。そして、検査装置において例えばアシンクロナスパケットの送受信用アプリケーションを立ち上げる。そして、図8に示すようなアプリケーション画面上で、キーボードから任意の1つのパケットのヘッダー、ブロックデータ、及びサイズ等を入力して送信すると、検査対象機器から受信したレスポンスパケットがアプリケーション画面上に表示される。また、送受信したACKデータや、受信アシンクロナスパケットカウンタ、バスリセット発生カウンタ、そしてバスリセット発生時にはセルフIDパケットが受信パケット画面上に表示される。

**【0005】** これにより、所望のパケットを送り、正しいレスポンスパケットが返ってきたかどうかをチェックすることによって、1394プロトコルを満たしているかどうかを検査することができる。また、送受信したACKデータや、受信カウンタ、バスリセット発生の有無等も合わせて検査することによって、より詳細で確実な検査を行うことができる。

**【0006】**

**【発明が解決しようとする課題】** しかしながら、前記の従来の方法では、1394プロトコルを満たしているかを検査するために膨大な量のパケットをキーボードから入力して送受信することが必要であった。また、結果が正しいかどうか人間が判断しなければならず、大変な労力が必要であった。

**【0007】** 本発明はこのような問題点を鑑みてなされ

たものであって、1394プロトコルを満たしているかどうかの検査を自動的に行えるようにした1394プロトコル検査方法及び装置を提供することを目的とする。

【0008】

【課題を解決するための手段】本発明に係る1394プロトコル検査方法は、1394シリアルバス対応の機器のプロトコルを検査する方法であって、少なくともこのプロトコルの検査に必要な送信アシンクロナスパケットとそれに対応する受信アシンクロナスパケットをデータファイルとして保有する検査装置を用意し、この検査装置と1394シリアルバス対応の機器を1394シリアルバスで接続し、検査装置の検査アプリケーション上で前記データファイルを開き、このデータファイルに書かれている送信アシンクロナスパケットを検査装置から1394シリアルバス対応の機器へ1394シリアルバスを介して順次送信し、1394シリアルバス対応の機器から返ってきた受信アシンクロナスパケットと前記データファイルに書かれている前記受信アシンクロナスパケットを比較することを特徴とするものである。

【0009】本発明に係る1394プロトコル検査装置は、1394シリアルバス対応の機器のプロトコルを検査する装置であって、少なくともこのプロトコルの検査に必要な送信アシンクロナスパケットとそれに対応する受信アシンクロナスパケットを有するデータファイルを備え、検査アプリケーション上で前記データファイルを開き、前記データファイルに書かれている送信アシンクロナスパケットを1394シリアルバス対応の機器へ1394シリアルバスを介して順次送信し、1394シリアルバス対応の機器から返ってきた受信アシンクロナスパケットと前記データファイルに書かれている前記受信アシンクロナスパケットを比較することを特徴とするものである。

【0010】本発明では、1394プロトコルの検査に必要な送信アシンクロナスパケットとそれに対応する受信アシンクロナスパケット等をデータファイルとして作成しておき、データファイル上に書かれている送信アシンクロナスパケットを順に送信しながら、データファイル上に書かれている受信アシンクロナスパケットと実際に受信したアシンクロナスパケットとの比較を自動的に行う。

【0011】

【発明の実施の形態】以下本発明の実施の形態について図面を参照しながら詳細に説明する。

【0012】図1に本発明を適用したシステムの構成を示す。このシステムはPCシステム1のPC本体2とカメラ一体型ビデオテープレコーダ（以下CAMという）5との間を1394シリアルバス6で接続したものである。そして、PCIバス対応のIEEE1394インターフェースボード（以下1394 I/Fボードと略す）4をPC本体2のPCIスロットに挿入し、PC本

体2上でプロトコル検査用のアプリケーションを作成し、アシンクロナスパケットを送受信することによって、所望の機器（ここではCAM）のIEEE1394プロトコルを検査することを実現した例である。

【0013】図2は図1のPC本体2の内部の構成の概略を示すものである。ここで、図1と同一の部分には図1に付した番号と同一の番号が付してある。

【0014】PC本体2の内部に設けられた1394 I/Fボード4には、物理層コントロールブロック（PHY）11と、リンク層コントロールブロック（LINK）12とが設けられている。

【0015】物理層コントロールブロック11は1394バスの初期化やバスの使用権の調停等を行う。また、リンク層コントロールブロック12との間で、各種制御信号の通信を行うとともに、これらの信号を1394シリアルバス6に対して送受信する。

【0016】リンク層コントロールブロック12は、パケットの作成／検出、誤り訂正処理等を行う。

【0017】PCシステム1の本体2の内部には、さらにCPU14と、RAM15と、モニターインターフェース16と、ファイルメモリ17と、アプリケーションメモリ18とが設けられている。

【0018】CPU14はPC本体2の全体の制御等を行う。RAM15はCPU14が各種データの処理を行う際のワークエリアとなる。モニターインターフェース16はモニター3との間の制御信号の通信やモニター3に対するビデオ信号の送信を行う。ファイルメモリ17は1394プロトコルの検査に必要な検査データのファイル及び検査結果のファイル等を格納する。アプリケーションメモリ18は検査用アプリケーション等を格納する。ファイルメモリ17及びアプリケーションメモリ18は、実際にはハードディスク装置の記憶エリアの一部として構成される。

【0019】RAM15上のアプリケーションプログラムは物理層コントロールブロック11とリンク層コントロールブロック12の制御、コマンドやレスポンスの作成等の処理を行う。このアプリケーションプログラムはコマンドやレスポンスを作成するときにはリンク層コントロールブロック12内に設けられたレジスタの所定のアドレスにデータを書き込む。また、他の機器が送信したコマンドやレスポンスは、前記レジスタの所定のアドレスに書き込まれた後、アプリケーションプログラムにより読み出される。

【0020】なお、実際にはPC本体2内には、さらにキーボードインターフェースやROM等が設けられているが、ここでは省略した。

【0021】本実施の形態ではファイルメモリ17の内部には、検査対象である機器（ここではCAM5）が1394プロトコルを満たしているかどうかを検査するために必要な一連の送信パケットとそれに対応する受信ア

シンクロナスパケット、アック（ACK）コード、所望のメッセージを促すメッセージコード、所望の場所で待時間を入れるウェイトコード、所望の場所でルート／非ルートになるようにバスリセットを起こすことができるバスリセットコード等がデータファイルとして格納してある。そして、検査用アプリケーションを立ち上げ、処理を開始すると、このデータファイルが開かれ、以後各種パケットの送信、アックの受信、エラーの検出等が自動的に実行される。

【0022】図3はPC本体2上で作成した検査用アプリケーションの処理を示すフローチャートである。まずPC本体2において検査用アプリケーションを立ち上げ、検査用データファイルを開く（ステップS1）。この時、モニター3の画面に表示される映像を図4に示す。

【0023】この画面のツールバー上でGOを選択すると（ステップS2）、データファイル上のデータを全て読み出し、文法エラーがないかチェックする（ステップS3）。ここで、文法エラーがあった場合には、データファイルにエラーがあることをメッセージ等でユーザーに知らせ、検査には入らないようにする。

【0024】次にデータファイルに格納されている検査項目の数をカウントする（ステップS4）。ここではカウント値がNであったものとする。

【0025】次のステップS5では後述するデータ処理カウンタ（図示せず）の値がNになっているかどうかを判断し、NになるまでステップS6の処理に移行する。そして、ステップS6ではデータファイルを順番に読み出し、そこに書かれている処理を実行する。さらに、データファイルから読み出した処理を実行する毎にデータ処理カウンタ（図示せず）の値を1インクリメントし、ステップS5に戻る。ここでは、図4に示すように、まずデータファイル上の送信を示すコード（ここではTX）の後に続くパケットを送信する。次にアックデータを示すコード（ここではATTACK）があるため、実際に受信したアックとファイル上のデータを比較して間違っていたら実際に受信したアックを次の行に挿入しエラーカウンタ（図示せず）を1インクリメントする。次は受信を示すコード（ここではRX）が現れるため、実際に受信したパケットとファイル上のデータを比較して間違っていたら実際に受信したパケットを次の行に挿入し、エラーカウンタを1インクリメントする。次にまたTXが現れるため、その後に続くパケットを送信する。

【0026】以下同様に、データファイル上のN個のデータを全て読み出して処理するまでアシンクロナスパケットの送受信を繰り返す、1394プロトコルの検査を行なう。

【0027】全データ分の検査が終了したら、モニターインターフェース16を介してモニター3の画面上に検査データと共にエラーの数を表示し、さらにセーブを促

すダイアログを表示する（ステップS7）。そして、画面のツールバー上でセーブのコマンドが入力されたら、ファイルメモリ17にテキストファイルとして保存する（ステップS8）。

【0028】ここで、必要に応じて、データファイル上にメッセージを示すコードを入れておけば「再生してください」等のメッセージボックスを表示したり、自分がルート（root）になるように（ここではBR\_ROOT）又は検査対象機器がルートになるようにバスリセットを起こす（ここではBR\_NOTROOT）意味のコードを入れておけば、所望のバスリセットを起こすこともできる。待ち時間を入れる意味のコード（ここではWAIT）を入れておけば待ち時間を入れることもできる。

【0029】図5は結果がOKの場合の検査結果データの例を示す。これはCSR（Control and Status Register）のノードIDレジスタを検査したものである。そして、図6及び図7は結果がNGの場合の検査結果データの例の前半部と後半部を示す。これはCSRのステートクリアレジスタを検査したものである。図7の「; RESULT RX」の後に続く「FFC00060, . . . 00000001」は受信したレスポンスパケットを示す。そして、その1行上の「RX」と「FFC10060, . . . 00000001」はデータファイルに書いてあるパケットで本来受信すべきパケットである。これらの検査結果データはモニター3の画面に表示された後、ファイルメモリ17にテキストファイルとして保存される。

【0030】なお、図4、図6～図7ではモニター3の画面上にはデータファイルの内容を“TX” “00000140 . . .”等のコードで表示したが、これらをメッセージ（このコードの場合は、“送信”と“パケットの宛先”等）で表示してもよい。

【0031】このように本実施の形態によれば、簡単操作かつ短時間で1394プロトコルを自動的に検査できる。また、検査装置を操作するユーザーは、プロトコルを知らなくても検査できる。さらに、検査結果はテキストファイルとして保存できるため、履歴を残したりデバッグなどに非常に有効に使える。また、検査用データの文法エラーもチェックできるため、プロトコルを正しく検査でき、かつ、ソースは書き換えなくても、データを書き換えるだけで任意のパケットの送受を行えるため、プロトコル及びセット仕様の変更等にも柔軟に対応できる。

【0032】

【発明の効果】以上詳細に説明したように、本発明によれば、自動的にアシンクロナスパケットの送受信を行うだけでなく、1394プロトコルとしてのパケットの正誤チェックまで行うため、検査のための操作が簡単になり、かつ検査時間を非常に短縮できる。また、1394

プロトコルを知らないユーザーでも簡単操作でプロトコルの検査ができる。さらに、検査結果はテキストファイルとして保存できるため、履歴を残したりデバッグなどに非常に有効に使える。また、データファイル上の検査データの書き換えだけで様々なプロトコルの検査に柔軟に対応できる。

【図面の簡単な説明】

【図1】本発明を適用したシステムの構成を示す図である。

【図2】図1のPC本体の内部の概略構成を示すブロック図である。

【図3】図1のPC本体上で作成した検査用アプリケーションの処理を示すフローチャートである。

【図4】図3の検査用アプリケーションを立ち上げ、検

査用データファイルを開いた時に図1のモニターの画面に表示される映像を示す図である。

【図5】結果がOKの場合の検査結果データの例を示す図である。

【図6】結果がNGの場合の検査結果データの例の前半部を示す図である。

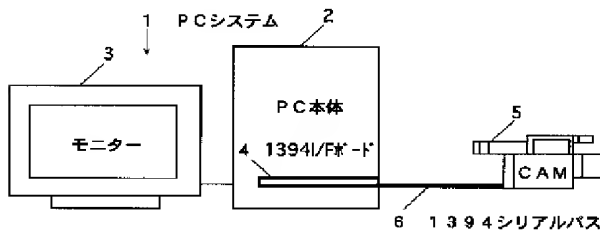
【図7】結果がNGの場合の検査結果データの例の後半部を示す図である。

【図8】従来のアシンクロナスパケットの送受信用アプリケーションの画面を示す図である。

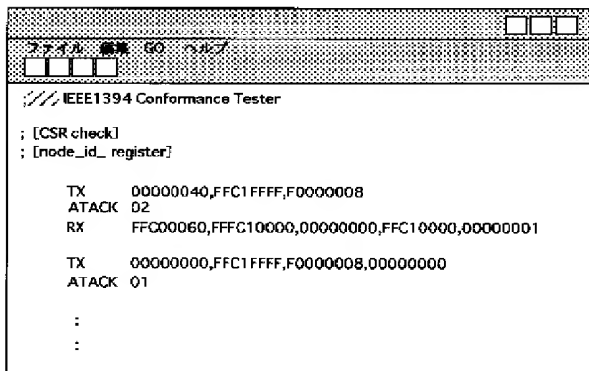
【符号の説明】

1…PCシステム、2…PC本体、4…1394 I/Fボード、5…CAM、6…1394シリアルバス、17…ファイルメモリ、18…アプリケーションメモリ。

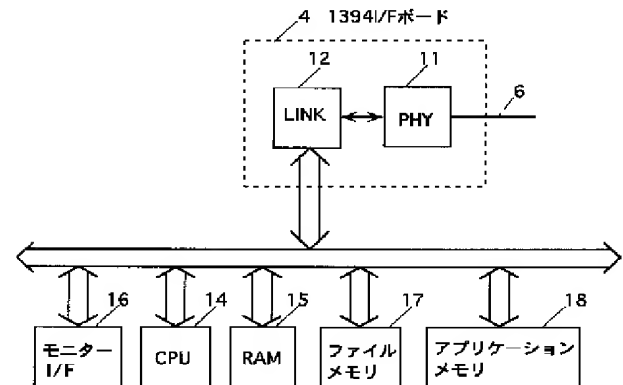
【図1】



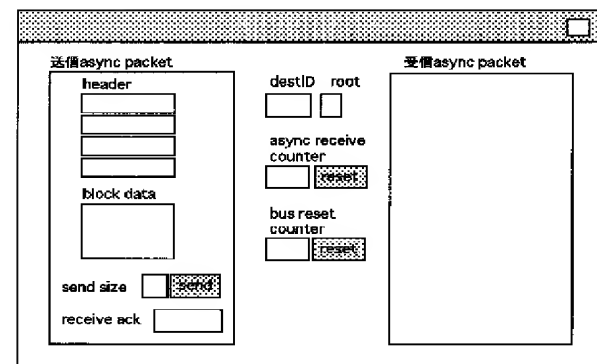
【図4】



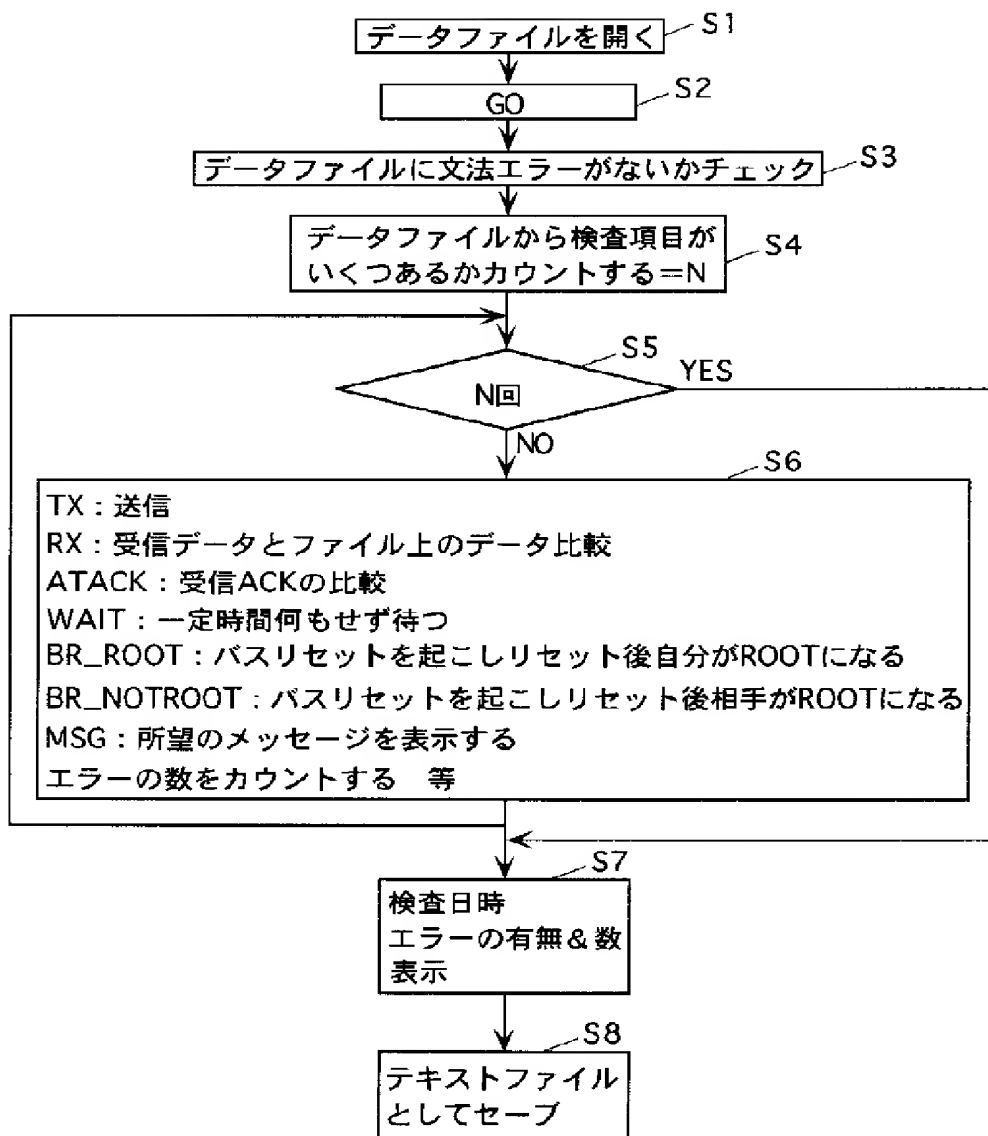
【図2】



【図8】



【図3】



【図5】

```

;///// IEEE1394 Conformance tester
; [CSR check]
; [NODE_IDS register]
; 検測機がroot
    BR_NOTROOT
    WAIT
    TX          00000140,FFC1FFFF,F0000008
    ATACK      02
    RX          FFC00160,FFC10000,00000000,FFC10000,00000001
    TX          00000100,FFC1FFFF,F0000008,00000000
    ATACK      01
    TX          00000140,0001FFFF,F0000008
    ATACK      02
    RX          FFC00160,00010000,00000000,00010000,00000001
    :
    中略
    :
    TX          00000190,0001FFFF,F0000008,00080002,FFC10000,FFFFFFF
    ATACK      02
    RX          FFC001B0,00010000,00000000,00040002,00010000,00000001
    TX          00000140,FFC1FFFF,F0000008
    ATACK      02
    RX          FFC00160,00010000,00000000,00010000,00000001
; -----
; [1394 Conformance Tester ] ;Test date: 1996/09/12 20:13:54
;
; No error.

```

【図6】

```

;///// IEEE1394 Conformance tester
; [CSR check]
; [STATE_CLEAR register]
; 検測機がroot
    BR_NOTROOT
    WAIT
    TX          00000040,FFC1FFFF,F0000000
    ATACK      02
    RX          FFC00060,FFC10000,00000000,00000100,00000001
    TX          00000040,FFC1FFFF,F0000000
    ATACK      02
    RX          FFC00060,FFC10000,00000000,00000100,00000001
    :
    中略
    :
    TX          00000090,FFC1FFFF,F0000000,00080002,00000100,FFFFFFF
    ATACK      02
    RX          FFC000B0,FFC10000,00000000,00040002,00000100,00000001
    WAIT
    TX          00000040,FFC1FFFF,F0000000
    ATACK      02
    RX          FFC00060,FFC10000,00000000,00000100,00000001
;

```

【図7】

```

; 検討機がrootでないとき (自分: PCがroot)
    BR_ROOT
    WAIT
    TX
    ATTACK 02      00000040,FFC0FFFF,F0000000
    RX
;RESULT RX        FFC10060,FFC00000,00000000,00000000,00000001
                    FFC00060,FFC10000,00000000,00000100,00000001
    TX
    ATTACK 01      00000000,FFC0FFFF,F0000000,00000000
    RX
;RESULT RX        FFC10060,FFC00000,00000000,00000000,00000001
                    FFC00060,FFC10000,00000000,00000100,00000001
    :
    中絶
    :
    BR_ROOT
    WAIT
    TX
    ATTACK 02      00000040,FFC0FFFF,F0000000
    RX
;RESULT RX        FFC10060,FFC00000,00000000,00000000,00000001
;-----
; [1394 Conformance Tester ] ;Test date: 1996/09/12 20:13:54
    15 erros

```